Combined Use of Prioritized AIMD and Flow-Based Traffic Splitting for Robust TCP Load Balancing*

Onur Alparslan, Nail Akar, and Ezhan Karasan

Electrical and Electronics Engineering Department,
Bilkent University, Ankara 06800, Turkey

Abstract. In this paper, we propose an AIMD-based TCP load balancing architecture in a backbone network where TCP flows are split between two explicitly routed paths, namely the primary and the secondary paths. We propose that primary paths have strict priority over the secondary paths with respect to packet forwarding and both paths are rate-controlled using ECN marking in the core and AIMD rate adjustment at the ingress nodes. We call this technique "prioritized AIMD". The buffers maintained at the ingress nodes for the two alternative paths help us predict the delay difference between the two paths which forms the basis for deciding on which path to forward a new-coming flow. We provide a simulation study for a large mesh network to demonstrate the efficiency of the proposed approach in terms of the average per-flow goodput and byte blocking rates.

Keywords: Traffic engineering; load balancing; multi-path routing; TCP.

1 Introduction

IP Traffic Engineering (TE) controls how traffic flows through an IP network in order to optimize the resource utilisation and network performance [4]. In multi-path routing-based TE, multiple explicitly routed paths with possibly disjoint links and nodes are established between the two end points of a network in order to optimize the resource utilisation by intelligent traffic splitting. These explicitly routed paths are readily implementable using standard-based layer 2 technologies like ATM or MPLS or using source routed IP tunnels. The work in [5] proposes a dynamic multi-path routing algorithm in connection-oriented networks where the shortest path is used under light traffic conditions and as the shortest path becomes congested, multiple paths are used upon their availability in order to balance the load. Recently, there have been a number of multi-path TE proposals specifically for MPLS networks that are amenable to distributed

* This work is supported in part by the Scientific and Technical Research Council of Turkey (Tübitak) under project EEEAG-101E048.

J. Solé-Pareta et al. (Eds.): QofIS 2004, LNCS 3266, pp. 124–133, 2004.

online implementation. In [7], the ingress node uses a gradient projection algorithm for balancing the load among the Label Switched Paths (LSP) by sending probe packets into the network and collecting congestion status. Additive Increase/Multiplicative Decrease (AIMD) feedback algorithms are used generally for flow and congestion control in computer and communication networks [6]. The multi-path AIMD-based approach of [17] uses binary feedback information for detecting the congestion state of the LSPs and a traffic splitting heuristic using AIMD is proposed in [17] which ensures that source LSRs do not send traffic to secondary paths of longer length before making full use of their primary paths.

Some multi-path routing proposals cause possible de-sequencing (or reordering) of packets of a TCP flow. This is due to sending successive packets of a TCP flow over different paths with different one-way delays. The majority of the traffic in the current Internet is based on TCP and this packet de-sequencing adversely affects the application-layer performance of TCP flows [10]. In order to avoid packet de-sequencing in multi-path routing, a flow-based splitting scheme that operates on a per-flow basis can be used [16]. In [14], flow-based multi-path routing of elastic flows are discussed. Flow-based routing in the QoS routing context in MPLS networks is described in [11], but the flow awareness requirement inside the core network may cause scalability problems with increasing number of instantaneous flows.

Recently, a new scalable flow-based multi-path TE approach for best-effort IP/MPLS networks is first proposed in [2] which employs max-min fair bandwidth sharing using an explicit rate control mechanism. This approach imposes flow awareness only at the edges of an MPLS backbone. This work demonstrates the performance enhancements attained by the flow-based splitting approach using comparisons with packet-based (i.e., non-flow based) multi-path routing and single-path routing when streaming traffic (i.e., UDP) is used. Significant reductions in packet loss rates are obtained relative to single-path routing in all the scenarios tested. This architecture is then studied for load balancing of elastic traffic (i.e., TCP) with AIMD-based rate control (as opposed to explicit rate for the sake of practicality) using a simple three node topology [3]. It is shown in [3] that flow-based multi-path routing method consistently outperforms the case of single-path. In the current paper, we provide an extensive simulation study of the approach proposed in [3] for TCP load balancing in larger and realistically sized mesh networks.

It is well-known that using alternative longer paths by some sources force other sources whose min-hop paths share links with these alternative paths to also use alternative paths [13]. This fact is called the knock-on effect in the literature and is studied in depth for alternately routed circuit switched networks [9]. Precautions should be taken to mitigate the knock-on effect for example the well-known "trunk reservation" concept in circuit switched networks [9]. One of the key ingredients of our proposed architecture is the use of strict priority queuing that favors packets of primary paths (PP) over those of secondary paths (SP) to cope with the knock-on effect. In this paper, we also compare and

contrast strict priority queuing with the widely deployed FIFO queuing in their capabilities to deal with the knock-on effect in the TCP load-balancing context.

The remainder of the paper is organized as follows. In Section 2, we present our TE architecture. We provide our simulation results in Section 3. The final section is devoted to conclusions and future work.

2 Architecture

This section is mainly based on [3] but the proposed architecture is outlined here for the sake of completeness. In this study, we envision an IP backbone network which consists of edge and core nodes (i.e., routers) and which has mechanisms for establishing explicitly routed paths. In this network, edge (ingress or egress) nodes are gateways that originate/terminate explicitly routed paths and core nodes carry only transit traffic. Edge nodes are responsible for per-egress and per-class based queuing, flow classification, traffic splitting, and rate control. Core nodes support per-class queuing and Explicit Congestion Notification (ECN) marking. In this architecture, flow awareness requirement is restricted to edge nodes making the overall architecture scale better than some other flow-based architectures.

Our architecture is based on the following building blocks: (i) queuing in network nodes, (ii) path establishment, (iii) feedback mechanism and rate control, and (iv) traffic splitting. As far as queuing is concerned, the core nodes employ per-class queuing with three drop-tail queues, namely the gold, silver, and bronze queues and strict priority queuing with the highest (lowest) priority given to the gold (bronze) queue, The gold queue is used for Resource Management (RM) and TCP ACK . We envision that ACK packets are identified by the ingress node and the encapsulation header for such packets are marked accordingly. Silver and bronze queues are used for TCP data packets according to the selection of paths as explained below. We assume in this study that edge nodes are single-homed, i.e., they have a link to a single core node. We setup one PP and one SP from an ingress node to every other egress node. We impose that the two paths are link-disjoint within the scope of the core network. The PP is first established as the min-hop path. If there are multiple min-hop paths, the one with the minimum propagation delay is chosen as the PP. In order to find the route for the SP, we prune the links used by the PP and compute the min-hop path in the remaining network graph. A tie in this step is broken similarly. If the connectivity is lost after the first step, we do not establish an SP. We prefer to use this simple path selection scheme since we do not assume a-priori knowledge of the traffic demand matrix.

In this paper, we study two queuing models based on the work in [2]. The first one is FIFO (first-in-first-out) queuing in which all the TCP data packets join the silver queue irrespective of the type of path they ride on. However, this queuing policy triggers the knock-on effect due to the lack of preferential treatment to packets using fewer resources (i.e., traversing fewer hops). Using longer secondary paths by some sources may force other sources whose primary

Table 1. The AIMD algorithm

if RM packet marked as CE
$\quad$ ATR := ATR − RDF × ATR
else
$\quad$ ATR := ATR + RIF × PTR
ATR := min(ATR, PTR)
ATR := max(ATR, MTR)

paths share links with these secondary paths to also use secondary paths. In order to mitigate this cascading effect, longer secondary paths should be resorted to only if primary paths can no longer accommodate additional traffic. Based on the work described in [2] and [3], we propose strict priority queuing in which TCP data packets routed over PPs use the silver service and those routed over SPs receive the bronze service.

Another building block of the proposed architecture is the feedback mechanism and rate control. In our proposed architecture, ingress nodes periodically send RM packets to egress nodes, one over the PP (P-RM) and the other over the SP (S-RM). These RM packets are sent in every T_{RM} seconds with the direction bit set to indicate the direction of flow. If strict priority queuing is used and when an P-RM (S-RM) packet arrives at the core node on its forward path, the node compares the percentage queue occupancy of its silver (bronze) queue on the outgoing interface with a predetermined configuration parameter μ and it sets the CE (Congestion Experienced) bit (if not already set) of the P-RM (S-RM) packet accordingly. If FIFO queuing is used then it is the silver queue occupancy that needs to be checked for both P-RM and S-RM packets. When an RM packet arrives at the egress node, it is sent back to the ingress node after resetting the direction bit of the RM packet. RM packets travelling over the reverse path are not marked by the core nodes. When the RM packet arrives back at the ingress node, the CE bit indicates the congestion status of the path it was sent over. According to the information, the ingress node updates the Allowed Transmission Rate (ATR) of the corresponding rate-controlled path by using the AIMD algorithm given in Table 1 [6]. In this algorithm, MTR and PTR denote the Minimum Transmission Rate and Peak Transmission Rate and RDF and RIF denote the Rate Decrease Factor and Rate Increase Factor, respectively. Therefore, an ingress node maintains two per-egress queues, one for the PP and the other for the SP, that are drained using AIMD-based rate control. The proposed architecture is depicted in Fig. 1 for an example 3-node network in which solid lines are for PPs whereas the dotted lines stand for SPs originating at ingress node 0. We also assume that the switching technology in the core network has the necessary fields in the encapsulation header for implementing the above-mentioned mechanisms.

The final ingredient to the proposed approach is the way we split traffic over the PP and the SP. The edge nodes first identify new flows. The delay estimates for the PP and SP queues (denoted by D_{PP} and D_{SP}, respectively) in the edge

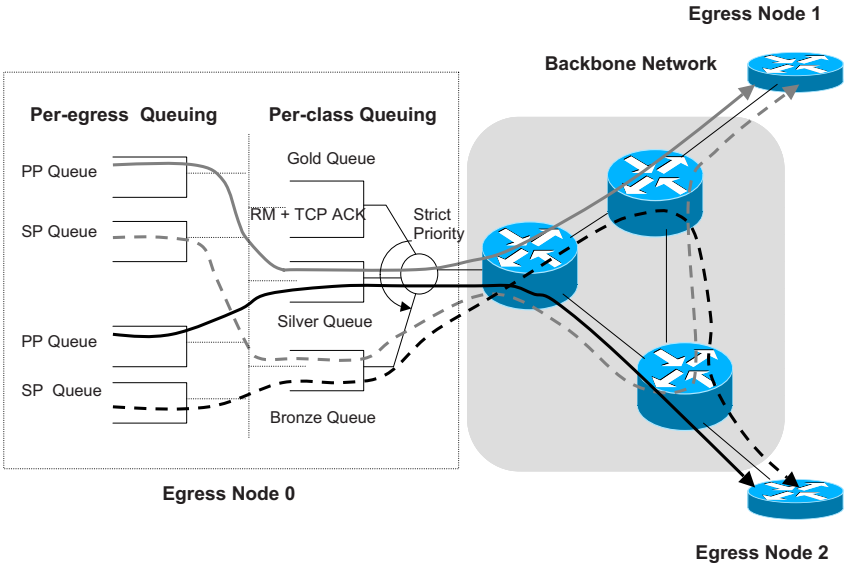

Fig. 1. The proposed architecture for an example 3-node network

nodes are then calculated by dividing the occupancy of the corresponding queue with the current drain rate. Upon the arrival of the first packet of the nth flow (i.e., a TCP SYN segment) a running estimate of the delay difference (denoted by d_n) is calculated as $d_n = \beta(D_{PP} - D_{SP}) + (1-\beta)d_{n-1}$, where β is the smoothing parameter. If $d(n) \leq min_{th}$ ($d(n) \geq max_{th}$) then we forward the flow over the PP (SP). When $min_{th} < d_n < max_{th}$, then the new flow is forwarded over the SP with probability $p_0(d_n - min_{th})/(max_{th} - min_{th})$ where min_{th}, max_{th} and p_0 are the splitting algorithm parameters to be set. In this paper, we use $p_0 = 1$. Once a path decision is made for the first packet of a flow, all the remaining packets of the flow will follow the same path. This traffic splitting mechanism is called Random Early Reroute (RER) which is inspired by the RED (Random Early Detect) algorithm used for active queue management in the Internet [8]; note the similarity in the algorithm parameters. RED is used for controlling the average queue occupancy whereas the average smoothed delay difference of silver and bronze queues is controlled by RER. RER parameters are generally chosen so that the PP is favoured (i.e., $min_{th} \geq 0$) and proportional control (as opposed to on-off control) is used, i.e., $max_{th} > min_{th}$.

3 Simulation Study

In this paper, we present the simulation results of our AIMD-based multi-path TE algorithm for TCP traffic over a mesh network called the hypothetic US topology that has 12 POPs (Point of Presence). This network topology and the traffic demand matrix are given in `www.fictitious.org/omp` and also

described in [2]. The proposed TCP TE architecture is implemented over ns-2 (Network Simulator) version 2.27 [12] and TCP-Reno is used in our simulations. We introduced a number of new modules and modifications in ns-2 that are available in [1].

In our simulations, we scaled down the capacities of all links and the demand matrix by a factor of 45/155 (replace all OC-3 links with DS-3) to reduce the simulation run-times. We assume that each of the POPs has one edge node connected via a very high speed link to one core node. We use a traffic model where flow arrivals occur according to a Poisson process and flow sizes have a bounded Pareto distribution denoted by $BP(k, p, \alpha)$ [15]. The following parameters are used for the bounded Pareto distribution in this study: $k = 4000$ Bytes, $p = 50 \times 10^6$ Bytes, and $\alpha = 1.20$, corresponding to a mean flow size of $m = 20,362$ Bytes. The delay averaging parameter is set to $\beta = 0.3$. TCP data packets are assumed to be 1040 Bytes long and RM packets are 50 Bytes long (after encapsulation). All the buffers at the edge and core nodes including per-egress (primary and secondary) and per-class queues (gold, silver and bronze), have a size of 104,000 Bytes each. The TCP receive buffer is of length 20,000 Bytes. We fix the following parameters for the AIMD algorithm. PTR is chosen as the speed of the slowest link on the corresponding path. We use very small but nonzero MTR in order to eliminate cases causing division by zero in the simulations. If the expected delay of a buffer exceeds 0.36 s, then the packets destined to the corresponding queue are dropped. We use $T_{RM} = 0.02$ s and $\mu = 20\%$. The simulation runtime is selected as 300 s. We report only the statistics related to those flows that have been initiated in the interval [90 s, 250 s].

We compare and contrast three TE policies using simulations. *Shortest path routing* policy uses the minimum-hop path with the AIMD-ECN capability turned on and there is no traffic splitting. The second TE policy is the *Flow-based Multi-path with Shortest Delay (SD) and FIFO queuing*. In this policy, SD refers to the specific RER setting $min_{th} = max_{th} = 0$ and therefore SD forwards each flow to the path with the minimum estimated queuing delay at the ingress edge node and it does not necessarily favour the PP. Moreover, we use SD in conjunction with the FIFO queuing discipline where there is no preferential treatment between the PP and the SP at the core nodes. The third TE policy is the *Flow-based Multi-path with RER and Strict Priority queuing* approach proposed in this paper.

The goodput of the TCP flow i (in bit/s), denoted by G_i, is defined as the service rate received by flow i during its lifetime. Mathematically, $G_i = \Delta_i / T_i$, where Δ_i is the number of bits successfully delivered to the application layer by the TCP receiver for flow i and T_i is the sojourn time of the flow i within the simulation runtime. We note that if flow i terminates before the end of the simulation, then Δ_i will be equal to the flow size S_i. One performance measure we study is the normalized average goodput defined as

$$G = \frac{\sum_i \Delta_i G_i}{\sum_i \Delta_i}.$$

However, we note that some flows are not fully carried due to overloading of certain links in the network. In order to take this effect into account, we introduce a new performance measure, called the net average goodput, denoted by G_{net}

$$G_{net} = \frac{\sum_i \Delta_i G_i}{\sum_i S_i},$$

by means of equating the service rate of un-carried packets to zero. For the same effect, we suggest a new measure, called the Byte Rejection Ratio (BRR), to quantify the portion of data that cannot be delivered within the simulation duration, in percentage. Mathematically,

$$\text{BRR} = \frac{\sum_{s,d} N(s,d) - \sum_{s,d} \Gamma(s,d)}{\sum_{s,d} N(s,d)} * 100,$$

where $N(s,d)$ is the sum of the sizes of flows demanded from node s to node d, and $\Gamma(s,d)$ is the total traffic (in bytes) successfully delivered to the application layer from node s to node d.

We first study the role of AIMD parameterization on the proposed TE in terms of G_{net} and BRR. Figures 2(a) and 2(b) demonstrate the effect of RIF and RDF on G_{net}. Similarly, Figures 2(c) and 2(d) present the effect of these AIMD parameters on BRR. In these simulations, RER parameters are chosen as $min_{th} = 1$ ms and $max_{th} = 15$ ms. We observe that flow-based multi-path with RER and strict priority queuing gives better performance in both measures than shortest-path routing. The choice of RDF= 0.0625 and RIF=0.0625 gives relatively good and robust performance in terms of G_{net} and therefore we use these parameters in the rest of the paper.

The effect of RER parameters on G_{net} and BRR are presented in Figures 3(a) and 3(b), respectively. We observe that the performance of the RER is quite robust except for the choices of RER parameters close to $min_{th} = max_{th} = 0$, i.e., the SD policy. We observe a sharp decline in the performance of the system when we apply the SD policy due to the induced knock-on effect. The simulation results show that G_{net} for the multi-path routing policy with RER and Strict Priority satisfies $G_{net} \geq 5.50$ Mbit/s when the RER parameters are in the range $0 \leq min_{th} \leq 1$ ms and 1 ms $\leq max_{th} \leq 15$ ms. For the same example, G_{net} is given by $G_{net} \approx 5.24$ Mbit/s and $G_{net} \approx 3.90$ Mbit/s for the shortest-path routing policy with and without AIMD, respectively. This shows that for a wide operational range for RER, multi-path routing policy outperforms single-path routing policies and the performance of the RER converges to that of the shortest-path routing policy with AIMD as we increase min_{th} and max_{th}. Based on these observations, we choose the RER parameters as $min_{th} = 1$ ms and $max_{th} = 15$ ms from this wide operational range.

Finally, we scale the incoming traffic by multiplying the flow sizes with a scaling parameter γ where $0.5 \leq \gamma < 1$ while fixing the flow arrival times. We then vary γ to see its impact on network performance. In Fig. 4(a), the multi-path TE with strict-priority and RER is shown to achieve the highest G_{net} for all values of γ. It is also observed from Fig. 4(a) that the proposed TE approach

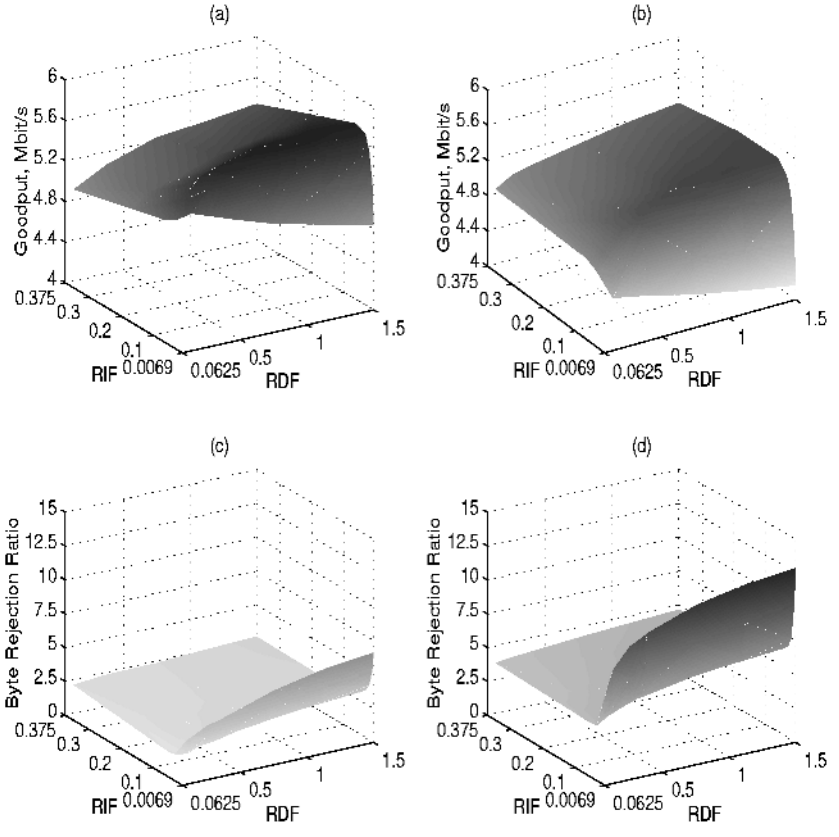

Fig. 2. As a function of RIF and RDF: (a) G_{net} for the multi-path TE with strict-priority and RER, (b) G_{net} for the shortest-path routing, (c) BRR for the multi-path TE with strict-priority and RER (d) BRR for the shortest-path routing

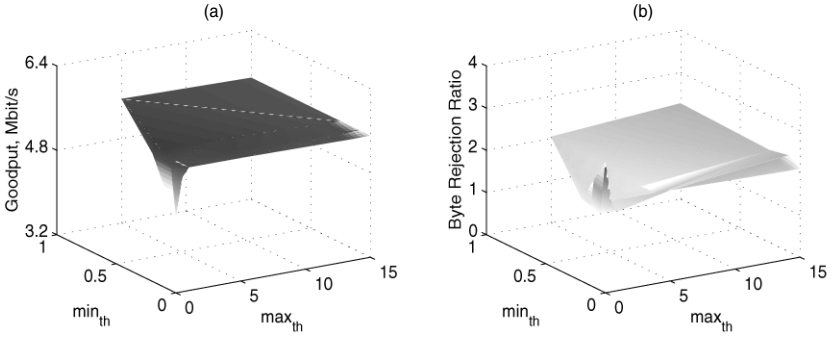

Fig. 3. As a function of min_{th} and max_{th}: (a) G_{net} for the multi-path TE with strict-priority and RER (b) BRR for the multi-path TE with strict-priority and RER

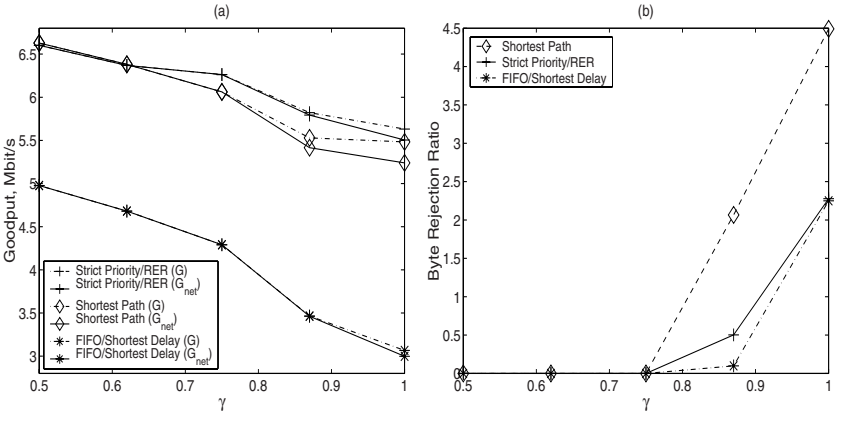

Fig. 4. As a function of traffic scaling parameter γ: (a) G_{net} and G (b) Byte Rejection Ratio

outperforms the other policies in terms of G as well. This shows that the multi-path TE with strict-priority and RER not only carries more traffic but also the carried flows are transported faster.

In Fig. 4(b), we observe that the policy of multi-path routing with strict-priority and RER has a BRR which is approximately half of that of the shortest-path routing policy for $\gamma = 1$. As the offered traffic decreases, the gap between the multi-path routing with strict-priority and RER and the shortest-path routing disappears. This is due to the fact that the PP is not congested at light traffic loads and the multi-path routing nearly boils down to shortest-path routing. We also observe that the SD routing with FIFO queuing gives lower BRR than the proposed TE policy for some values of γ. However, the net goodput of the multi-path routing with SD and FIFO queuing is 25-50% lower than the proposed TE approach when γ varies between 0.5 and 1.0, as shown in Fig. 4(a).

4 Conclusions

In this paper, we report our findings on a recently proposed TCP load balancing architecture that uses prioritized AIMD and flow-based multi-path routing with RER. Using a publicly used test network, we show that our proposed architecture consistently outperforms the case of a single path in terms of average normalized goodput and the byte rejection ratio. We show in this paper that the architecture stays robust for relatively large networks, extending our existing results for small topologies. On the other hand, we also show that employing load balancing with conventional FIFO queuing and shortest delay policies does not always produce better results than that of a single path, which can be explained by the knock-on effect. Future work in this area will consist of incorporating a-priori knowledge on the traffic demand matrix into the proposed architecture.

References

1. The multi-path routing project at Bilkent University Information Networks Laboratory (BINLAB). Web page: `http://www.binlab.bilkent.edu.tr/onur/index.html`, July 2004.
2. N. Akar, I. Hokelek, M. Atik, and E. Karasan. A reordering-free multipath traffic engineering architecture for Diffserv/MPLS networks. In *Proceedings of IEEE Workshop on IP Operations and Management*, pp. 107–113, Kansas City, Missouri, USA, 2003.
3. O. Alparslan, N. Akar, and E. Karasan. AIMD-Based Online MPLS Traffic Engineering for TCP Flows via Distributed Multi-Path Routing. www.binlab.bilkent.edu.tr/e/journal/125/annales_final.pdf. Accepted for publication in Annales Des Telecommunications.
4. D. O. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao. Overview and principles of Internet traffic engineering. IETF Informational RFC-3272, May 2002.
5. S. Bahk and M. E. Zarki. Dynamic multi-path routing and how it compares with other dynamic routing algorithms for high speed wide area networks. In *ACM SIGCOMM*, pp. 53–64, 1992.
6. D. M. Chiu and R. Jain. Analysis of the increase/decrease algorithms for congestion avoidance in computer networks. *Computer Networks and ISDN Systems*, 17(1):1–14, June 1989.
7. A. Elwalid, C. Jin, S. Low, and I. Widjaja. MATE: MPLS Adaptive Traffic Engineering. In *Proceedings of INFOCOM*, pp. 1300–1309, 2001.
8. S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, 1993.
9. F. P. Kelly. Routing in circuit switched networks: Optimization, shadow prices and decentralization. *Advances in Applied Probability*, 20:112–144, 1988.
10. M. Laor and L. Gendel. The effect of packet reordering in a backbone link on application throughput. *IEEE Network Magazine*, 16(5):28–36, 2002.
11. Y-D. Lin, N-B. Hsu, and R-H. Hwang. QoS routing granularity in MPLS networks. *IEEE Comm. Mag.*, 46(2):58–65, 2002.
12. S. McCanne and S. Floyd. ns Network Simulator. Web page: `http://www.isi.edu/nsnam/ns/`, July 2002.
13. S. Nelakuditi, Z. L. Zhang, and R. P. Tsang. Adaptive proportional routing: A localized QoS routing approach. In *Proceedings of INFOCOM*, Anchorage, USA, 2000.
14. S. Oueslati-Boulahia and J. W. Roberts. Impact of trunk reservation on elastic flow routing. In *Networking 2000*, March 2000.
15. I. A. Rai, G. Urvoy-Keller, and E. W. Biersack. Analysis of LAS scheduling for job size distributions with high variance. In *Proceedings of ACM Sigmetrics*, pp. 218–228, 2003.
16. A. Shaikh, J. Rexford, and Kang G. Shin. Load-sensitive routing of long-lived IP flows. In *ACM SIGCOMM*, pp. 215–226, 1999.
17. J. Wang, S. Patek, H. Wang, and J. Liebeherr. Traffic engineering with AIMD in MPLS networks. In *7th IFIP/IEEE International Workshop on Protocols for High-Speed Networks*, pp. 192–210, 2002.